34(Currently amended). A method of dynamically configuring and emulating a hardware architecture of a processing system that processes one or more nodes representing a solution space for a problem, via a computer system with an operating system, to determine a solution for said problem, wherein at least one of said nodes includes data for said problem, said method comprising:

(a) emulating said hardware architecture to implement a virtual machine, via said computer system, and managing said nodes within said solution space, wherein said hardware architecture is designed to process said nodes;

(b) configuring said hardware architecture, via a virtual operating system, and controlling said emulation of said hardware architecture based on a user software application defining said problem and corresponding processing to determine said solution, wherein step (b) further includes:

(b.1) configuring said nodes of said solution space in a topology to determine said solution for said problem via a configuration engine of said virtual operating system and storing and evaluating said data for said problem within said nodes via a population engine of said virtual operating system, wherein said nodes are created and deleted via an instantiation engine of said virtual operating system;

(b.2) traversing said topology and processing selected ones of said nodes based on said user software application to determine said solution via a navigation engine of said virtual operating system; and

(b.3) updating said nodes and said topology based on said user software application via an evolution engine of said virtual operating system.

35(Currently amended). The method of claim 34, wherein step (b.1) further includes:

(b.1) generating and sending instructions from said virtual operating system to said virtual machine to create and delete said nodes, to store and evaluate said data within said nodes, and to store parameters defining said topology of said nodes representing said problem and configuration parameters for said hardware architecture;

step (b.2) further includes:

(b.2.1) generating and sending instructions from said virtual operating system to said virtual machine to traverse said topology and retrieve information from said selected nodes; and

step (b.3) further includes:

(b.3.1) generating and sending instructions from said virtual operating system to said virtual machine to update said nodes and said topology.

36(Currently amended). The method of claim 35, wherein step (b) further includes:

(b.4) converting said instructions from said virtual operating system into a format compatible with said virtual machine via a virtual assembler of said computer system.

37(Currently amended). The method of claim 34, further including:

(c) interfacing said virtual machine with said computer system by converting commands for said virtual machine into commands compatible with said computer system via a platform driver unit of said computer system.

38(Currently amended). The method of claim 34, wherein step (a) further includes:

(a.1) creating one or more additional instances of said processing system that each include subsets of said emulated hardware architecture via a thread unit of said virtual operating system, wherein each instance is associated with a corresponding task to enable said tasks to be performed concurrently.

39(Currently amended). The method of claim 34, wherein said computer system includes an arithmetic logic unit, and step (a) further includes:

(a.1) emulating an arithmetic logic unit of said hardware architecture via said computer system, and selectively passing arithmetic operations for said emulated hardware architecture to said arithmetic logic unit of said computer system.

40(Previously presented). The method of claim 34, wherein said computer system includes at least one of local and distributed networks of processing systems, and said virtual machine includes a corresponding instruction set.

41(Previously presented). The method of claim 34, wherein each node includes an index word and a data word.

42(Previously presented). The method of claim 34, wherein each node includes one or more of numeric tags, character tags, boolean flags, numeric values, character values, object

identifications, database-record identifications, simple arrays, variable-density multidimensional arrays, symbolic functions, mathematical functions, connection pointers to other nodes, function pointers, lookup-table list pointers, linked-lists, and pointers to other solution spaces, data representations, procedures, or other virtual machines.

43(Previously presented). The method of claim 34, wherein said topology includes at least one of independent point-clouds, ordered sets of points, acyclic graphs, cyclic graphs, balanced trees, recombining graphs, meshes, and lattices.

44(Previously presented). The method of claim 39, wherein said emulated arithmetic logic unit provides fixed-point integer arithmetic with precision indicated by said user software application.

45(Currently amended). The method of claim 40, wherein step (b) further includes:

(b.4) managing distribution of data and processes to said networked processing systems via a network unit of said computer system.

46(Currently amended). The method of claim 34, wherein step (b) further includes:

(b.4) managing daemons for background processing of said nodes via a process unit of said virtual operating system; and

(b.5) enabling performance of frequently-used tasks via a toolbox unit of said virtual operating system.

47(Previously presented). The method of claim 34, wherein said hardware architecture includes a non-Von Neumann architecture.

48(Previously presented). The method of claim 34, wherein said hardware architecture includes a reduced instruction set computer (RISC) architecture.

49(Previously presented). The method of claim 34, wherein said emulation of said hardware architecture includes employing at least one of a small instruction set, simple and efficient data representation and handling, inherent vector representation, limited data/calculation modes, interleaved memory, table lookup, induced pointers, and distributed and parallelized computation.

50(Currently amended). The method of claim 34, wherein step (b.2) further includes:

(b.2.1) pre-computing said data within said nodes of said solution space via a computation module of said computer system to enable navigation of possible solutions to occur in near real-time.

51(Previously presented). The method of claim 46, wherein said daemons operate concurrently to perform tasks including at least one of collecting garbage, pruning trees, condensing redundancies, processing edit-queues, interpolating with finer granularity around selected nodes in said solution space, and extrapolating and elaborating said data during processing and navigation of said nodes.